

基於深度學習的驗證碼自動識別方法研究

組員: 1100423 陳叡逸 1100426 林冠宇

摘要

隨著網路應用的普及，驗證碼（CAPTCHA）在保障網路安全中扮演重要角色，但其手動解讀需求阻礙了自動化測試的效率。本文提出了一種基於深度學習的自動驗證碼識別方法，結合卷積神經網絡（CNN）與門控循環單元（GRU）構建的卷積循環神經網絡（CRNN）架構，有效提升了複雜驗證碼的識別準確率。為減少對大量標註數據的依賴，我們採用了分類、預訓練與微調相結合的策略。首先在低複雜度數據集上進行預訓練，再利用少量新類型數據微調，最終用一個簡單 CNN 模型將同類型的驗證碼導向相應類的微調模型。實驗結果表明，CRNN 模型在字元和字串準確率上均優於 CNN 模型，且 ResNet-18 結合 GRU 的架構表現最佳。該方法不僅提高了驗證碼識別的準確性和效率，還具備良好的擴展性，提供一個快速可靠自動化測試方式。

關鍵詞：驗證碼識別、深度學習、卷積神經網絡（CNN）、門控循環單元（GRU）、預訓練與微調、網路安全

目錄

1. 引言.....	4
2. 文獻回顧.....	4
2.1 驗證碼識別技術.....	4
2.2 深度學習在驗證碼識別中的應用.....	4
3. 研究方法.....	6
3.1 模型架構.....	6
3.1.1 驗證碼分類 (CNN).....	6
3.1.2 驗證碼識別 (CRNN).....	6
3.1.3 損失函數 (CTC Loss).....	7
3.2 CRNN_GRU 與 ResNet-18 之架構比較與改進.....	8
3.2.1 相似性分析.....	10
3.2.2 關鍵改進與調整.....	10
3.2.3 最終架構對比.....	13
3.3 預訓練與微調.....	14
3.4 資料集準備.....	14
3.4.1 高複雜度數據集 - 辨識能力測試.....	15
3.4.2 低複雜度數據集 - 預訓練.....	15
4. 實驗設計.....	15
4.1 模型效能比較.....	15
4.1.1 CNN 與 CRNN 模型比較.....	15
4.1.2 CRNN 模型比較.....	15
4.2 低複雜度驗證碼預訓練.....	16
4.3 少量新類型驗證碼微調.....	16
4.4 驗證碼分類.....	16
5. 實驗結果.....	17
5.1 模型辨識能力.....	17
5.1.1 CNN vs CRNN 比較.....	19
5.1.2 CRNN 模型比較.....	19
5.2 預訓練.....	20
5.3 使用預訓練模型訓練少量新類型驗證碼.....	20

5.4 驗證碼分類.....	23
5.5 預測.....	23
6. 結論.....	24
參考資料.....	26

1 引言

隨著網路應用的普及，網路安全問題日益突出。驗證碼（CAPTCHA）作為一種有效區分人類與自動程序的技術，廣泛應用於防止惡意攻擊，如暴力破解密碼和自動化腳本。驗證碼的自動化特性使其在降低人力成本和提升可靠性方面具有優勢，對於增強網路安全性具有不可忽視的作用。然而，對於軟體測試人員而言，驗證碼的手動操作需求成為自動化測試流程中的一大障礙，影響測試效率。

基於此背景，本文旨在利用深度學習在圖像識別領域的優勢，提出一種新穎的自動識別複雜驗證碼的方法。與傳統依賴圖像分割技術的方法不同，我們採用了由卷積神經網絡（CNN）與循環神經網絡（RNN）組成的模型，並通過實驗驗證其在驗證碼識別準確性優勢。此外，考慮到深度學習模型通常需要大量標註資料進行訓練，本文提出了一種基於少量標註資料的預訓練與分類策略，旨在降低資料標註的時間與成本，提升模型的泛化能力。

2 文獻回顧

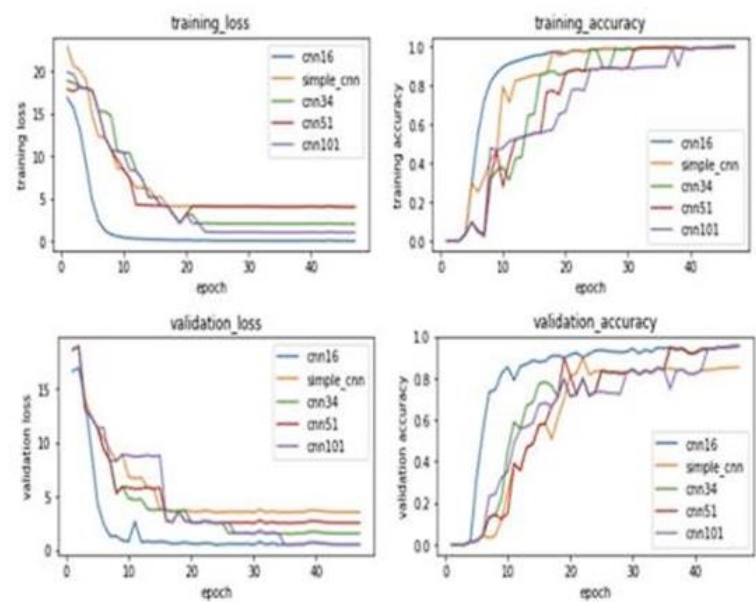
2.1 驗證碼識別技術

傳統的驗證碼識別方法多依賴於圖像分割技術，將驗證碼圖像分割成單個字元進行識別。然而，隨著驗證碼複雜度的增加，字元間的扭曲和重疊使這種方法的效果逐漸下降。近年來，深度學習技術在圖像識別領域取得了顯著進展，尤其是卷積神經網絡（CNN）在特徵提取方面展現出強大的能力。

2.2 深度學習在驗證碼識別中的應用

多項研究已經將 CNN 與循環神經網絡（RNN）結合應用於驗證碼識別。例如 Yuan（2018）中研究了不同深度的 CNN 模型（3 層、16 層、34 層、51 層和 101 層）對於驗證碼識別的效果，發現 16 層的 CNN 在辨識效果上表現最佳，隨著層數的增加，辨識效果並未進一步提升。最終，該研究選用了 CNN 與 GRU（門控循環單元）結合的架構，並指出當遇到與訓練資料集中不同類型的

驗證碼時，辨識效果會下降，但通過少量的新增訓練資料，可以有效提升模型的泛化能力。



該文獻最終使用以下架構 CNN+GRU

Name	Category	Filter size/ stride/ padding	Channel	output size
Conv1	Convolution	3x3/1/1	64	64x100x300
Conv2	Convolution	3x3/1/1	64	64x100x300
Pool1	Max Pooling	2X2/2	64	64x50x150
Conv3	Convolution	3x3/1/1	128	128X50X150
Conv4	Convolution	3x3/1/1	128	128X50X150
Pool2	Max Pooling	2X2/2	128	128x25x75
Conv5	Convolution	3x3/1/1	256	256x25x75
Conv6	Convolution	3x3/1/1	256	256x25x75
Conv7	Convolution	3x3/1/1	256	256x25x75
Conv8	Convolution	3x3/1/1	256	256x25x75
Pool3	Max Pooling	2X2/2	256	256x12x37
Conv9	Convolution	3x3/1/1	512	512x12x37
Conv10	Convolution	3x3/1/1	512	512x12x37
Conv11	Convolution	3x3/1/1	512	512x12x37
Conv12	Convolution	3x3/1/1	512	512x12x37
Pool4	Max Pooling	4x4/4	512	512x3x9
Conv13	Convolution	3x3/1/1	512	512x3x9
Conv14	Convolution	3x3/1/1	512	512x3x9
Conv15	Convolution	3x3/1/1	512	512x3x9
Conv16	Convolution	3x3/1/0	512	512X1X7

3 研究方法

3.1 模型架構

本文提出的驗證碼識別系統由兩個主要部分組成：驗證碼分類和驗證碼識別。

3.1.1 驗證碼分類(CNN)

首先，我們設計了一個基於簡單 CNN 的驗證碼分類模型，用於預先分類驗證碼的類型（字體、扭曲、雜訊）。該分類模型通過三層卷積結構提取圖像特徵，並將驗證碼分類到對應的類別。分類結果將引導驗證碼進入對應的微調模型進行更精確的識別。

在這個系統中，分類模型的主要作用是 **將不同類型的驗證碼圖片對應到對應的辨識模型**。這種設計有以下幾個優勢：

1. 提高準確率：

- ✧ 不同類型的驗證碼可能具有不同的特徵和難度。通過專門對每種類型驗證碼訓練模型，可以針對性地提高每個類別的準確率。

2. 減少計算資源消耗：

- ✧ 如果所有驗證碼都使用同一個模型處理，模型需要具備更高的泛化能力，可能導致模型複雜度增加且更難訓練。而通過分類模型先將圖片分類，可以使用較為簡單且高效的專用模型。

3. 擴展性佳：

- ✧ 當有新的驗證碼類型出現時，只需訓練分類模型和對應的模型，就可快速適應新的類別。

3.1.2 驗證碼識別（CRNN）

在本研究中，驗證碼識別部分採用了卷積循環神經網絡（Convolutional Recurrent Neural Network，CRNN）架構。CRNN 結合了卷積神經網絡（CNN）

和循環神經網絡（RNN）的優點，既能有效提取圖像的空間特徵，又能捕捉序列數據中的時序依賴關係，特別適合處理需要序列預測的任務，如語音、文字識別等。

CRNN 的具體架構如下：

✧ **卷積層（CNN 部分）：**

- 使用 ResNet-18 作為基礎的卷積神經網絡結構。ResNet-18 具有跳躍連接（Skip Connections），能夠有效緩解深層網絡中的退化問題（Degradation Problem），提升特徵提取的效果。
- 卷積層負責提取驗證碼圖像的空間特徵，生成特徵圖（feature maps），這些特徵圖將作為後續 RNN 部分的輸入。

✧ **循環層（RNN 部分）：**

- 在卷積層之後，本模型接入門控循環單元（GRU）層。GRU 是一種改進的 RNN 結構，與長短期記憶網絡（LSTM）相比，GRU 擁有更少的參數和更高的計算效率。GRU 通過其更新門（Update Gate）和重置門（Reset Gate），能夠有效捕捉序列數據中的長期依賴關係，雖然在某些情況下 LSTM 在捕捉長期依賴上表現更佳，但 GRU 以其簡潔的結構在多數應用中提供了足夠的性能表現。
- 循環層負責處理卷積層輸出的序列特徵，捕捉字元之間的上下文訊息，從而提升整體識別的準確性。

✧ **全連接層與輸出層：**

- 最後，將 GRU 層的輸出通過全連接層進行轉換，並使用 CTC 損失函數進行訓練，對驗證碼字元序列預測。

本研究選擇了 ResNet-18 作為基礎的卷積神經網絡結構，微調參數後與 GRU 結合形成 CNN+GRU 架構。此外，亦考慮了與長短期記憶網絡（LSTM）的組合，即 ResNet-18+LSTM，並進行微調以適應驗證碼識別任務。

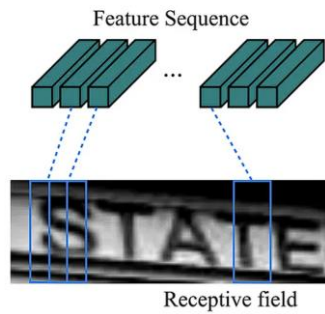
3.1.3 損失函數(CTC Loss)

在許多序列到序列的任務中，如將音頻轉換為文字或將驗證碼圖像轉換為字元序列，輸入序列（例如音頻幀或圖像特徵）和輸出序列（如文字）的長度通常不同，且缺乏明確的對齊關係。傳統的損失函數，如 Cross-Entropy Loss，需要輸入和輸出序列長度相同，並且需要精確對齊標註，這在實際應用中難以實

現。

CTC 通過引入一個空字元 (blank symbol) 和允許字元重複來解決這一問題，對於給定的輸入序列，CTC 會計算所有可能對應於目標輸出序列的對齊方式，並以空字元分割不同的輸出字元，將這些方式的概率總和作為最終的損失值。這樣，模型在訓練時無需知道每個字元具體應對輸入序列的哪一部分。

x1	x2	x3	x4	x6	x6	input(X)
C	O	[blank]	O	O	K	alignment
C	O	[blank]	O		K	output(Y)



在本研究中，驗證碼辨識將輸入的驗證碼圖像轉換為對應的字元序列。由於每個驗證碼包含多個字元，且字元之間可能存在扭曲、重疊或背景雜訊，導致字元的具體位置和邊界變得模糊不清。使用 CTC 損失函數具有以下優勢：

- ✧ **無需精確對齊標註：**CTC 不要求訓練數據中每個字元的具體位置，減少了數據標註的工作量和成本。
- ✧ **處理可變長度輸出：**驗證碼的字元數量可能變化，CTC 能夠靈活處理不同長度的輸出序列。
- ✧ **提升序列預測準確性：**通過考慮所有可能的對齊方式，CTC 能夠更準確地學習字元之間的依賴關係，提升整體識別準確率。

3.2 CRNN_GRU 與 ResNet-18 之架構比較與改進

本研究基於 ResNet-18 進行模型改進，以適應 OCR (Optical Character Recognition) 任務，並設計了一種結合卷積神經網絡 (CNN) 與門控循環單元 (GRU) 的 CRNN_GRU 模型。以下為改進後的完整模型架構，並對 ResNet-

18 與此模型之架構進行詳細比較，以及關鍵改動之理論依據。

Name	Category	Filter size / stride / padding	Input	Output Channels	Output
Input (C, H, W)	Input	-	(1, 32, 128)	-	(1, 32, 128)
Conv1	Convolution	3×3 / 1 / 1	(1, 32, 128)	64	(64, 32, 128)
Layer1 Block1	BasicBlock	3×3 / 1 / 1	(64, 32, 128)	64	(64, 32, 128)
Layer1 Block2	BasicBlock	3×3 / 1 / 1	(64, 32, 128)	64	(64, 32, 128)
Layer2 Block1	BasicBlock	3×3 / 2 / 1	(64, 32, 128)	128	(128, 16, 64)
Layer2 Block2	BasicBlock	3×3 / 1 / 1	(128, 16, 64)	128	(128, 16, 64)
Layer3 Block1	BasicBlock	3×3 / 2 / 1	(128, 16, 64)	256	(256, 8, 32)
Layer3 Block2	BasicBlock	3×3 / 1 / 1	(256, 8, 32)	256	(256, 8, 32)
Layer4 Block1	BasicBlock	3×3 / 2 / 1	(256, 8, 32)	512	(512, 4, 16)
Layer4 Block2	BasicBlock	3×3 / 1 / 1	(512, 4, 16)	512	(512, 4, 16)
AdaptiveAvgPool	Pooling	(1, W)	(512, 4, 16)	512	(512, 1, 16)
Squeeze & Permute	Operation	-	(512, 1, 16)	512	(16, 512)
Bi-GRU Layer 1	RNN (GRU)	-	(16, 512)	256×2=512	(16, 512)
Bi-GRU Layer 2	RNN (GRU)	-	(16, 512)	256×2=512	(16, 512)
Fully Connected	Linear	-	(16, 512)	num_classes	(16, num_classes)
Output	Final Output	-	(16, num_classes)	-	4-6 characters

3.2.1 相似性分析

CRNN_GRU 與 ResNet-18 在 CNN 結構上仍存在諸多相似之處，具體特性如下：

特性	ResNet-18	CRNN_GRU
基礎卷積結構	BasicBlock（殘差塊）	BasicBlock（殘差塊）
卷積層數	4 個殘差層	4 個殘差層
通道擴展	64 → 128 → 256 → 512	64 → 128 → 256 → 512
卷積核大小	3×3	3×3
批量正則化 （BatchNorm）	有	有
激活函數	ReLU	ReLU

此部分基本沿襲 ResNet-18 的設計，因其在影像特徵提取方面已展現優異性能。

3.2.2 關鍵改進與調整

為適應 OCR 及 CTC（Connectionist Temporal Classification）損失函數，本研究針對 ResNet-18 進行以下結構性改進：

(1) 首層卷積核縮小（7×7 → 3×3）

模型	第一層卷積
ResNet-18	7×7 卷積，stride=2
CRNN_GRU	3×3 卷積，stride=1

改動原因：ResNet-18 主要用於分類任務，適用於較大輸入影像（224×224），而驗證碼多為細長的文本影像（128×32），較大卷積核可能導致細節資訊流失。因此，本研究改用較小的 3×3 卷積，並將步長（stride）設為 1，以保留更多

局部特徵。

(2) 刪除池化層 (MaxPool)

模型	池化方式
ResNet-18	MaxPool(3×3, stride=2)
CRNN_GRU	刪除池化層

改動原因：ResNet-18 採用 MaxPool(3×3, stride=2) 進行特徵下採樣，以降低計算成本並提取更具判別力的特徵。然而，在 OCR 任務中，保持特徵圖的寬度 W 對於序列建模至關重要。因此，本模型移除 MaxPool 以維持 W 不變，並在 CRNN 最終階段引入 AdaptiveAvgPool(1, W)，將特徵圖高度 H 壓縮至 1，以更好地適配後續 RNN 模組的時序建模需求。

(3) 捨棄全連接層，加入 GRU 模型

模型	後處理方式
ResNet-18	512 維特徵 → 全連接層 → Softmax (分類)
CRNN_GRU	512 維特徵 → GRU (時序建模) → 全連接層 → CTC Loss

改動原因：ResNet-18 的全連接層適用於單一分類標籤的任務，而 OCR 任務涉及動態長序列輸出。因此，本模型引入雙向 GRU 以學習字元間的時序關係，使模型具備更強的序列建模能力。

(4) 時間步長 (Timestep) 計算與調整

在 OCR 任務中，時間步長 T 對應於 CNN 特徵提取後的影像寬度 W。由於 CNN 會對影像進行下採樣，因此 T 需經過計算：

$$T = (W - \text{kernel} + 2 * \text{padding}) / \text{stride} + 1$$

模型中， kernel size = 3 ,padding=1 代入以上公式，可簡化為

$$T = W / \text{stride}$$

根據本模型設計，最終模型的輸入維度為：

$$X_{\text{rnn}} \in \mathbb{R}^{B \times 16 \times 512} \text{ 其中, } T = 16$$

(5) 損失函數變更 (CrossEntropy \rightarrow CTC Loss)

模型	損失函數
ResNet-18	CrossEntropy Loss (適用於分類)
CRNN_GRU	CTC Loss (適用於序列學習)

改動原因：CTC Loss 可適應無對齊的序列標註，例如：

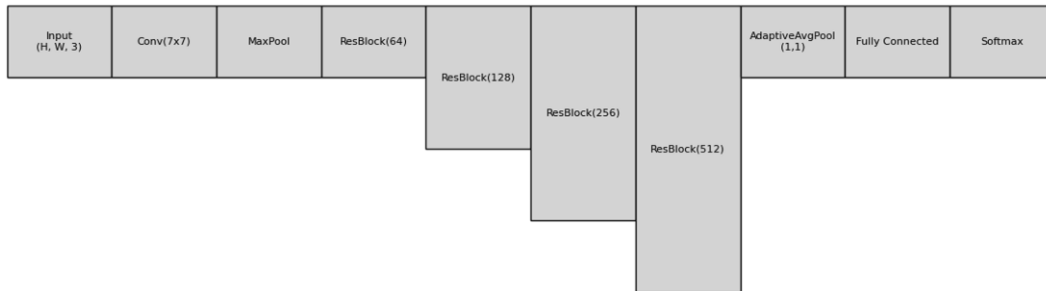
- 目標標籤："AA-AAB-B-BCC-CC"
- 模型輸出："AABBCC" (CTC 可自動合併重複字符並忽略空白 "-")

```
Timestep 0: [0, 0, 0, 1, 0] -> 'A'
Timestep 1: [0, 0, 0, 1, 0] -> 'A'
Timestep 2: [0, 0, 0, 0, 1] -> '-'
Timestep 3: [0, 0, 0, 1, 0] -> 'A'
Timestep 4: [0, 0, 1, 0, 0] -> 'B'
Timestep 5: [0, 0, 0, 0, 1] -> '-'
Timestep 6: [0, 0, 1, 0, 0] -> 'B'
Timestep 7: [0, 0, 0, 0, 1] -> '-'
Timestep 8: [0, 0, 1, 0, 0] -> 'B'
Timestep 9: [0, 0, 0, 0, 1] -> '-'
Timestep 10: [0, 0, 1, 0, 0] -> 'B'
Timestep 11: [0, 1, 0, 0, 0] -> 'C'
Timestep 12: [0, 1, 0, 0, 0] -> 'C'
Timestep 13: [0, 0, 0, 0, 1] -> '-'
Timestep 14: [0, 1, 0, 0, 0] -> 'C'
Timestep 15: [0, 1, 0, 0, 0] -> 'C'

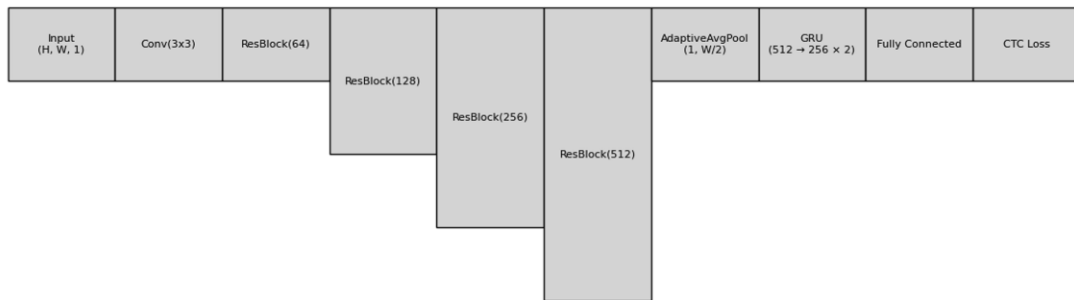
CTC Decoded sequence: AABBCC
```

3.2.3 最終架構對比

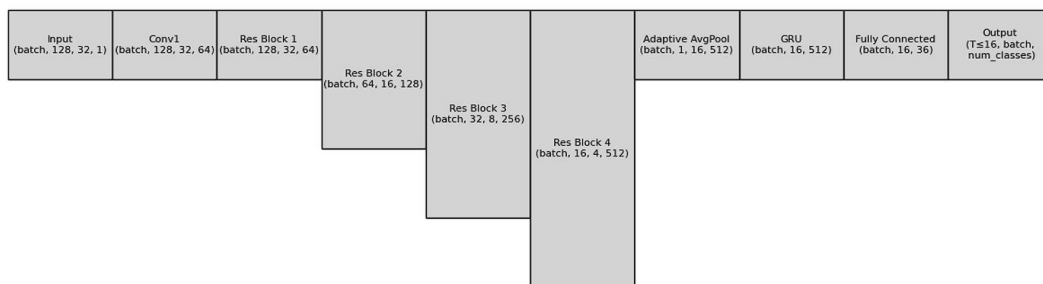
ResNet-18



ResNet-18_GRU



本研究輸入驗證碼的格式為 128x32x1(長 x 寬 x 灰階)，輸出為每個字元分為 36 類(0-9,a-z) ，4 到 6 位數組合，模型處理過程如下：



透過上述改進，本研究提出之 CRNN_GRU 模型具備以下優勢：

- 適應變長輸入與輸出：採用時間步長(Timestep)進行動態序列建模。
- 適用於無對齊數據：CTC Loss 可自動對齊輸出序列與標註。
- 強化時序建模能力：雙向 GRU 可捕捉字符間依賴關係。

- **適用於 OCR 任務：**適合應用於驗證碼識別、手寫文字識別等場景。

綜上所述，本研究透過對 ResNet-18 進行結構性調整，使其由傳統影像分類模型轉變為適用於 OCR 任務之時序識別模型，並驗證其效能與優勢。

3.3 預訓練與微調

為了適應不同類型的驗證碼，本研究採用了預訓練與微調相結合的策略。首先，利用低複雜度的驗證碼對模型進行預訓練，使模型能夠學習到基本的驗證碼特徵。接著，通過加入少量新類型的驗證碼數據對模型進行微調，以提升其對這些新類型驗證碼的識別能力。這種方法不僅加快了模型的訓練速度，還有效提升了模型在處理多樣化驗證碼時的準確性和泛化能力。

3.4 資料集準備

本研究使用了來自 kaggle 的驗證碼數據集來進行模型的訓練與評估，分別是高複雜度和低複雜度的驗證碼數據集，並對這兩個數據集進行了適當的預處理。

3.4.1 高複雜度數據集-辨識能力測試

	4 位數	3000 張
	5 位數	3000 張
	6 位數	3000 張
	5 位數	11000 張
		20000 張

高複雜度數據集由 4 至 6 位字大小寫英文字母與數字的隨機組合。這類驗證碼經常伴隨字元的扭曲、重疊以及雜訊設計，以防止傳統的圖像分割技術進行自動識別。我們將以此模型進行辨識能力測試。

3.4.2 低複雜度數據集-預訓練

2 WP 2	4 位數	10000 張
--------	------	---------

低複雜度數據集由固定 4 位的大小寫英文字母和數字隨機組合構成，字元之間只有有簡單的重疊、扭曲。我們將使用該數據集進行預訓練，能讓模型先掌握簡單的字元識別，為後續微調打下基礎。

4 實驗設計

實驗設計部分涵蓋了五個主要步驟，包括模型效能比較、低複雜度驗證碼的預訓練、少量新類型驗證碼的微調、驗證碼分類以及最終的預測流程。以下將進一步詳細說明每個步驟的具體實施細節。

4.1 模型效能比較

在比較實驗中，且所有模型均使用相同的訓練集和測試集，並且訓練回合數相同，以確保比較的公平性。評估指標包括字元準確率和字串準確率（即整個驗證碼字串完全正確的比例）。

4.1.1 CNN 與 CRNN 模型比較：

為了評估 CRNN 相較於 CNN 模型在驗證碼識別上的優勢，我們選取了高複雜度數據集中的約 10,000 張 5 位數驗證碼進行比較。

4.1.2 CRNN 模型比較：

在 CRNN 模型的比較實驗中，我們選擇了以下三種不同的 CRNN 架構進行評估：

- ✧ **ResNet-18+GRU**：本研究提出的主要模型，結合了 ResNet-18 的卷積特徵提取能力與 GRU 的序列建模能力。
- ✧ **ResNet-18+LSTM**：將 ResNet-18 與長短期記憶網絡（LSTM）結合，評估其在驗證碼識別上的表現。
- ✧ **CNN 16 layer+GRU**：參考文獻中使用的 16 層 CNN 結構與 GRU 結合的模型，作為對比基準。

4.2 低複雜度驗證碼預訓練：

為了提升模型在複雜驗證碼識別任務中的表現，我們首先使用低複雜度的驗證碼數據集（9,000 張）對 CRNN 模型進行預訓練。這些低複雜度驗證碼由固定 4 位的大小寫英文字母和數字隨機組合構成，字元之間僅存在簡單的重疊和少量雜訊。預訓練的目的是讓模型先掌握基本的字元識別能力，為後續的微調步驟打下基礎。

4.3 少量新類型驗證碼微調：

在完成預訓練後，我們引入少量的新類型驗證碼進行微調訓練。這些新類型的驗證碼具有更高的複雜度，包括高度的扭曲、重疊以及更多的背景雜訊。通過在預訓練模型的基礎上加入少量新類型的驗證碼數據進行微調，我們期望模型能夠快速適應新的驗證碼特徵，提升其在新類型驗證碼上的識別準確率。

4.4 驗證碼分類：

為了進一步提升整體系統的效率和準確性，我們設計了一個基於 CNN 的驗證碼分類模型。該分類模型負責根據驗證碼的特徵（如字體、扭曲程度、雜訊干擾等）將驗證碼分類到不同的類別，並引導驗證碼進入對應的 CRNN 識別模型。這一設計不僅提高了識別準確率，還有效降低了計算資源的消耗，並提升了系統的靈活性和擴展性。

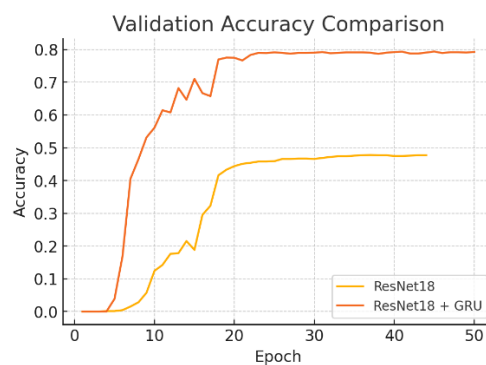
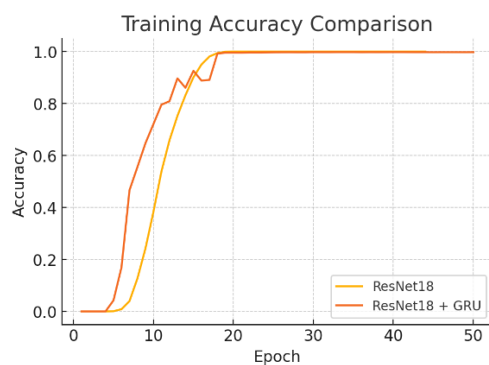
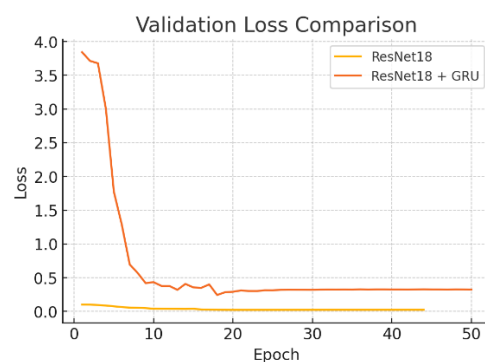
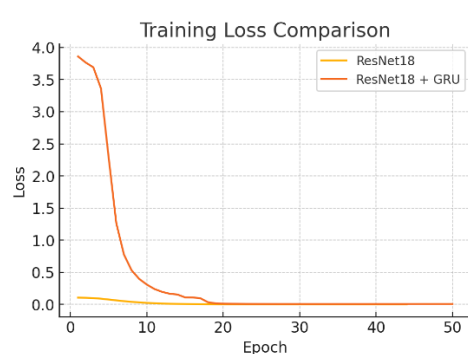
實驗結果

5.1 模型辨識能力

5.1.1 CNN vs CRNN 比較

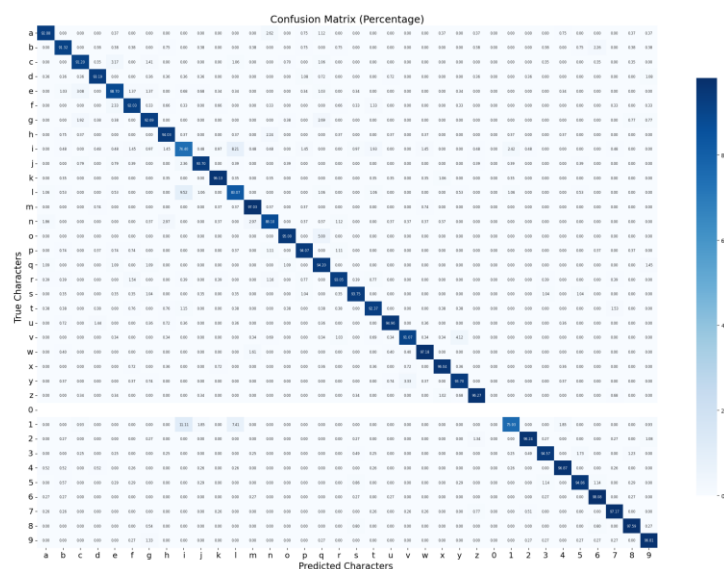
從高複雜度數據集中選取約 10000 張 5 位數驗證碼進行比較，結果如下：

	字元準確率	字串準確率(完全正確)
ResNet-18+GRU	97.79% (正確: 9309/9500)	80.02% (正確: 1525/1900)
ResNet-18	57.13% (正確: 5428/9500)	49.42% (正確: 939/1900)

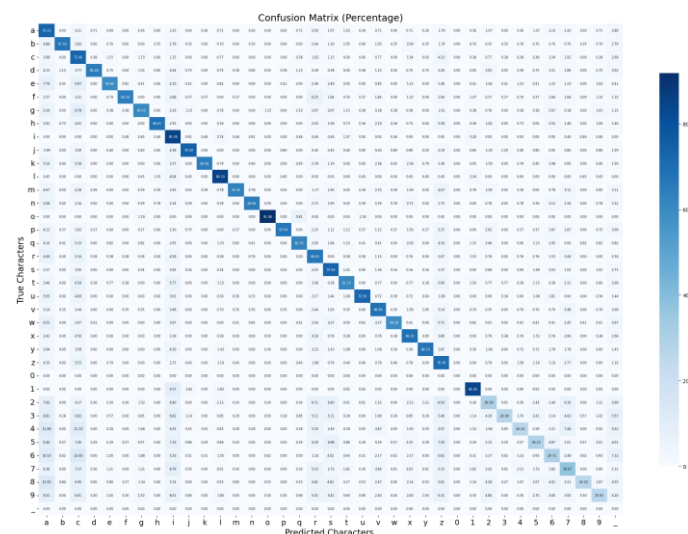


混淆矩陣比較

ResNet-18+GRU



ResNet-18



從測試集的結果和混淆矩陣中，我們可以觀察到 ResNet-18（僅使用 CNN）模型的字元準確率和字串準確率之間差異較小，但整體的辨識能力明顯低於 ResNet-18+GRU（CRNN）模型。這是因為：

1. **CNN 的局限**：ResNet-18 模型只能基於每個單獨字元的圖像進行預測，無法捕捉字元之間的上下文信息，這使得它容易將整個字串預測錯誤。

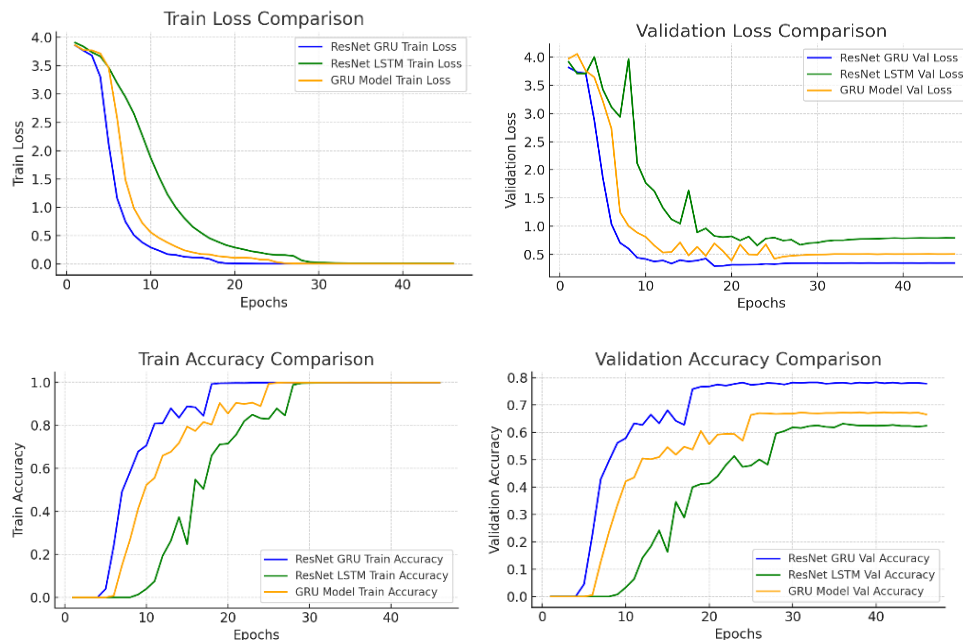
由於每個字元的預測都是相對獨立的，CNN 對於序列數據的表現相對較弱。

2. **CRNN 的優勢**：CRNN 結合了 CNN 和 GRU，能夠同時提取字元圖像特徵並處理字串中的時序依賴關係。它不僅能夠基於圖像特徵進行預測，還能捕捉字串中的上下文關聯性，因此 CRNN 對於整體字串的預測更準確。只有某些易混淆的字元會產生誤判(i 和 l、 m 和 n)，但它的序列處理能力能減少這些誤判對其他字元影響。

因此，即使在相同的訓練集、測試集和訓練回合數的情況下，CRNN 由於其更強的上下文處理能力，表現出了更好的泛化能力和更高的預測準確率。這說明 CRNN 在處理需要時序關聯的任務時，比單純的 CNN 模型更加適用。

5.1.2 CRNN 模型比較

	字元準確率	字串準確率(完全正確)
RESNET-18+GRU	93.45% (正確: 9309/9961)	76.56% (正確: 1525/1992)
RESNET18+LSTM	86.07% (正確: 8573/9961)	58.13% (正確: 1158/1992)
CNN 16 layer+GRU	89.77% (正確: 8942/9961)	65.16% (正確: 1298/1992)



這些結果的差異可能是由以下幾個因素導致的：

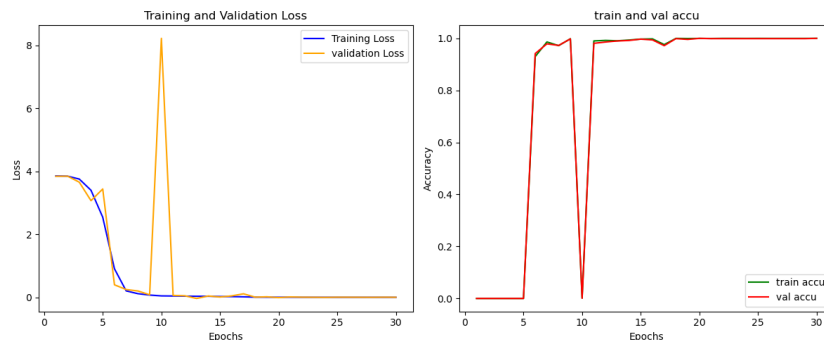
1. **ResNet 的跳躍連接 (Skip Connections)**：ResNet-18 利用了跳躍連接來避免退化問題，使模型能更好地傳遞前幾層學到的特徵資訊，從而增強了模型的穩定性和表現。因此，ResNet-18 + GRU 表現最佳，字元和字串的準確率都很高。

2. RNN 結構的差異 (GRU vs LSTM)：

驗證碼中的字元通常是短小且固定長度的序列，沒有太多長期依賴關係需要捕捉。LSTM 的長期記憶能力在此場景中可能並未完全發揮。GRU 的結構相對簡單，能夠更有效地處理這類短期依賴問題，因此能在驗證碼識別中表現更好。

5.2 預訓練


使用低複雜度的驗證碼數據集（9,000 張）進行預訓練後，模型在該類型驗證碼上的準確率已達到 99%，為後續的微調步驟打下了良好的基礎。



5.3 使用預訓練模型訓練少量新類型驗證碼

以下圖表展示了在使用預訓練模型和從頭開始訓練模型時，隨著訓練樣本數量增加，模型的識別準確率變化情況

使用預訓練模型訓練

	Training data	Accuracy(%)
	0	0
	200	41
	400	67

600	83
800	88
1200	94

從頭開始訓練

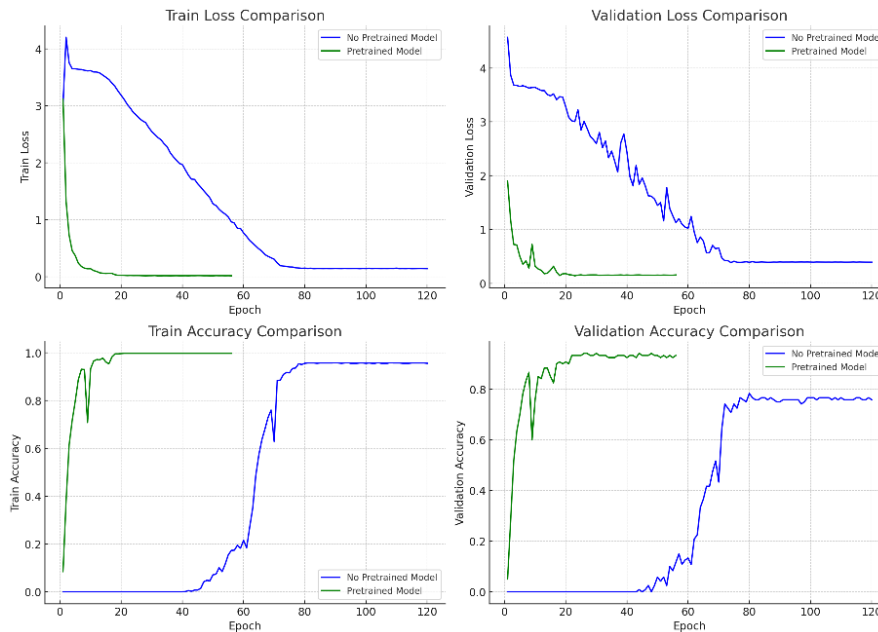


Training data	Accuracy(%)
600	0
800	40
1200	76
1400	84

辨識準確率/樣本數



訓練效率以 1200 樣本數為例：



分析結果

預訓練模型：

- ✧ **學習效率高**：即使在少量數據下（200 張），準確率已達 41%，隨著數據量增加，準確率迅速提升。
- ✧ **高準確率**：在 1200 張樣本時，準確率達 94%。
- ✧ **收斂速度快**：在第 20 epoch 模型已收斂達到最高狀態

從頭開始訓練：

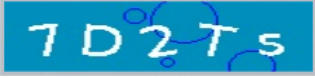
- ✧ **樣本需求高**：在 600 張樣本時，準確率仍為 0%，800 張時僅為 40%。
- ✧ **收斂速度慢**：在第 80 epoch 模型才收斂達到最高狀態


從比較結果中我們可以看出，預訓練策略在少量標註數據下顯著提升了模型的識別能力，且隨著數據量的增加，模型的準確率迅速提升，最終達到較高水平。


此外，對其他類型驗證碼的實驗結果也顯示，預訓練模型在不同數量訓練樣本下均能快速達到 90% 以上的識別準確率，具體數據如下：

其他類型驗證碼在不同數量訓練樣本的準確度差異

Training data	Accuracy(%)
---------------	-------------

	0	0
	100	40
	200	85
	400	100

	Training data	Accuracy(%)
	0	0
	100	26
	200	60
	400	87

	Training data	Accuracy(%)
	0	0
	100	50
	200	75
	400	90

這些結果進一步證明了預訓練策略在提升模型識別能力和泛化能力方面的優勢

5.4 驗證碼分類

在驗證碼分類部分，我們從每種類型的驗證碼中各選擇 100 張進行分類模型的訓練。最終，分類模型的準確度達到 100%，表明其在將驗證碼正確分類到相應類別方面具有極高的精確性。這為後續的 CRNN 識別模型提供了準確的分類依據，進一步提升了整體系統的識別效果。

5.5 預測

整個驗證碼識別系統的預測流程如下：

1. 驗證碼分類

- 使用分類模型預測待識別圖片屬於哪個驗證碼類型。
 - 根據分類結果，載入相應的識別模型。
2. 驗證碼預測：
- 使用對應的識別模型對驗證碼圖片進行內容預測，得到字元序列。
3. 結果保存：
- 將每張圖片的名稱、來源資料夾及預測的驗證碼內容寫入 predictions.csv 文件中，便於後續分析和應用。

預測結果

從五種類型的驗證碼中隨機選擇 10 張圖片進行預測，結果如下：

	Image	Source Folder	Predicted CAPTCHA
1	1BaXY.jpg	img-1	1baxy
2	1bbbF.jpg	img-1	1bbbf
3	1bBdK.jpg	img-1	1bbdk
4	1eqwk.jpg	img-1	1eqwk
5	1jKMP.jpg	img-1	1jkmp
6	2aJu2.png	img-4	2aju2
7	2i134.png	img-2	2i134
8	2i87.png	img-5	2j87
9	3g2w6.png	img-3	3g2w6
10	3x325.png	img-3	3x325

這些預測結果展示了系統在不同類型驗證碼上的識別能力，並且所有預測結果均正確。

6. 結論

在本研究中，我們提出一種基於深度學習的驗證碼自動識別方法，成功結合卷積神經網絡（CNN）與門控循環單元（GRU），形成具備時序處理能力的 CRNN 模型，以辨識複雜驗證碼。實驗結果顯示，CRNN 在高度扭曲或重疊的驗證碼情境下，於字元準確率與字串準確率方面均優於僅使用 CNN 的模型。

此外，本研究引入預訓練策略以降低模型對大量標註資料的依賴。我們先以低複雜度驗證碼資料集進行預訓練，學習基本字元特徵，再以少量新類型驗證碼

進行微調，提升模型的泛化能力與辨識效果。實驗結果證實，該策略能顯著提升模型在少量標註資料條件下的準確率，並展現良好的跨類型識別能力。

為進一步提升系統效率與準確性，我們設計一個基於 CNN 的驗證碼分類模型，能根據字體、扭曲程度與雜訊干擾等特徵將驗證碼歸類，並將分類結果導向對應識別模型以提升辨識精度。此分類機制有效降低運算資源消耗，搭配預訓練策略，使系統在面對新型驗證碼時僅需簡單微調即可快速適應。

綜上所述，本研究透過「分類機制」、「預訓練策略」與「模型微調」三者協同設計，成功實現多樣化驗證碼之高效分類與準確辨識，並提升系統在有限資料下的學習效率與實用表現，展現其於實務應用上的可行性與擴展性。未來研究可進一步探索更先進之網路架構與訓練策略，以因應更高複雜度與更多變化的驗證碼挑戰。

為促進研究之透明性與可重現性，本研究完整程式碼已公開於 GitHub，涵蓋資料前處理、模型訓練與分類架構等實作內容，網址如下：

<https://github.com/rayyichen310/Research-on-Automatic-CAPTCHA-Recognition-Method-Based-on-Deep-Learning>

參考資料

- [1] Yuan, Z.-Y. (2018). 運用深度神經網絡實現驗證碼識別 [Master's thesis, National Taiwan University]. NTU Repository.
- [2] Shi, B., Bai, X., & Yao, C. (2015). An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *arXiv preprint arXiv:1507.05717*.
- [3] Noury, Z., & Rezaei, M. (2020). Deep-CAPTCHA: A deep learning based CAPTCHA solver for vulnerability assessment. *arXiv preprint arXiv:2006.08296v2*.
- [4] qjadud1994. (2020). CRNN-Keras [Source code]. GitHub.
<https://github.com/qjadud1994/CRNN-Keras>
- [5] 老農的博客. (2021). 寫給程式設計師的機器學習入門(八) - 卷積神經網路 (CNN) - 圖片分類和驗證碼識別. <https://303248153.github.io/ml-08/>
- [6] Chen, R. (2024). *Research on Automatic CAPTCHA Recognition Method Based on Deep Learning* [Source code]. GitHub.
<https://github.com/rayvichen310/Research-on-Automatic-CAPTCHA-Recognition-Method-Based-on-Deep-Learning>

